

TCP-Westwood 针对卫星网的改进方案

黄 蕾, 刘立祥

(中科院软件所通用软件实验室, 北京 100080)

摘 要: TCP-Westwood (TCPW) 是针对无线网络误码率高的特点而设计的。它能在大误码率情况下取得较好的性能, 但它丢包较多、在长延时环境中性能下降较快, 这使得它不能直接在卫星网中使用。该文针对卫星网特点, 提出一种 TCPW 的改进方案: 每次带宽估计计算后确定当前拥塞窗口是否适宜, 一旦当前窗口大于估计带宽, 就迅速降低窗口。仿真实验证明, 这种改进方案不仅提高了 TCPW 的吞吐量, 而且使丢包数大幅度下降。

关键词: 拥塞控制; TCP; 卫星网

Improvement of TCP-Westwood over Satellite Networks

HUANG Lei, LIU Lixiang

(General Software Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing 100080)

【Abstract】 TCP-Westwood (TCPW) is shown to provide significant performance improvement in the case of high BER. But its performance gain is achieved on the base of packet losses and its throughput decreases with RTT growing. An improved scheme TCPW-S is proposed in this paper to make TCPW more suitable for satellite network with scarce resource and long delay. After every RTT, congestion window is adjusted to the estimated bandwidth with a conservative policy. Simulation results reveal that throughput of TCPW-S increases while packet losses drop a lot.

【Key words】 Congestion control; TCP; Satellite networks

1 概述

随着卫星组网技术的发展, 卫星网络逐渐成为 Internet 的承载网络。然而传统的地面网传输协议 TCP 并不适合卫星网络的特点。目前, 适用于卫星网络的传输层协议已成为世界各国的研究热点。

TCP-Westwood(TCPW)^[1]针对无线网络的特点对传统 TCP 的拥塞控制协议进行了改进。它提出了一种发送端的改进方案, 通过观测到达的 ACK 及其间隔时间, 确定网络的可用带宽。在分组丢失时, 用估算的可用带宽计算新的窗口和慢启动门限, 而不是像传统 TCP 那样盲目降低窗口。TCPW 针对无线网络错误率高的特性设计了有效的应对方法, 使其性能得到了大幅度提高。

卫星网是一种特殊的无线网络。除了误码率高的特点外, 卫星网的另一个主要特征是延时长。TCPW 虽然很好地处理了高误码率的影响, 但在长延时方面, 它并未涉及。仿真实验表明, 随着延时的增加, TCPW 性能下降的较快。究其原因, 是由于 TCPW 在拥塞避免阶段的窗口扩充策略与传统 TCP 相同, 只要收到 ACK, 窗口就会增加, 而不管实际可用带宽如何。这种处理方式使拥塞丢包成为必然。在延时较小时, TCPW 通过丢包后的处理可及时校准窗口值, 提高吞吐量。随着延时的增大, 上一 RTT 的测量值和当前实际网络情况的相关性下降, 校准的精度下降, 导致 TCPW 的性能下降。

本文对 TCPW 的算法进行改进, 使其能根据可用带宽及时降低窗口。改进 TCPW 在拥塞避免阶段窗口盲目增加的算法, 根据窗口和估计带宽的比较结果来确定窗口变化值。使拥塞即将发生时, 窗口能及时下降到适宜的水平, 而不是只有丢包时才进行窗口减小的操作。仿真实验证明, 这种更加保守的窗口改变方式提升了 TCPW 在大延时情况下的性能。

2 相关工作

相对于地面网络, 卫星网络具有延时长、带宽延时乘积大、误码率高、轨间链路延时持续变化、链路不对称等特点。传统的 TCP 协议(Reno)不适宜直接在卫星网络中使用。针对卫星网络的不同特点, 人们提出了各种的改进方式, 如增大初始窗口、延迟应答、多个流共享参数、使用 TCP 代理等^[2]。

适宜的拥塞控制机制是提高 TCP 性能的有效手段。当前针对卫星网络的拥塞控制机制的研究重点集中在两个方面: 如何区分包丢失原因以免将因链路错误而引起的丢包按照拥塞丢包处理? 如何估计网络的可用带宽使窗口调节更加准确? 这两个方面相辅相成。当前主流的端到端改进方式包括:

(1) TCP-Vegas^[3]提出了一种粗粒度的带宽估算方法, 根据当前吞吐量与预期的差值来确定窗口的增减方式。同时通过 DUPACK 的时间与发送时间的间隔来加快重传速度。

(2) TCP-Peach^[4]则通过低优先级分组来探测网络的闲置带宽, 加快慢启动和快速恢复期间窗口扩大的速度。

(3) TCP-Westwood(TCPW)提出了一种较为精确的带宽估计算法, 通过监控到达的 ACK 流确定可用带宽。一旦出现丢包, 就迅速把窗口恢复到与带宽相适应的水平, 从而消除了高误码率的影响。

这 3 种改进方案代表了 3 种不同的解决思路。人们吸取了这些思想的精华, 对他们进行更深入的研究, 并开发出各

基金项目: 国家“863”计划基金资助项目(2005AA712025)

作者简介: 黄蕾(1972-), 女, 博士生, 主研方向: 卫星网上的拥塞控制机制, 无线传感器网络的信任机制; 刘立祥, 博士、副研究员

收稿日期: 2006-06-07 **E-mail:** tqhlff@public3.bta.net.cn

种新的版本,如在TCP-Vegas基础上提出的AdaVegas^[5,6];在TCP-Peach基础上提出的TCP-Peach + ^[7]、TCP-Swift^[8];在TCPW基础上提出的TCPW-A^[9]等。本文在TCPW基础上,针对卫星网特性提出一种新的改进方案。

3 TCPW 分析

在TCPW中,发送方持续监控到达的ACK流,在此基础上估算网络当前的可用带宽。TCPW的动态估算系统如下:

$$BE_k = \alpha_k BE_{k-1} + (1 - \alpha_k) \frac{b_k + b_{k-1}}{2}, \quad \alpha_k = \frac{2\tau - \Delta t}{2\tau + \Delta t}, \quad b_k = \frac{d_k}{t_k - t_{k-1}} \quad (1)$$

其中 d_k 是第 k 个ACK所响应的数据量, t_k 是第 k 个ACK的到达时间, BE_k 是 t_k 时刻的带宽估计值, $1/\tau$ 是滤波器的截止频率。

检测到分组丢失后,TCPW将根据带宽估计来确定新的窗口和慢启动门限值。具体来说,如果收到3个DUPACK,TCPW就把窗口和门限改为

$$\begin{cases} ssthresh = (BE * RTT \min) / seg_size \\ cwin = \min(cwin, ssthresh) \end{cases} \quad (2)$$

如果TCPW检测到重传计时器超时,则新的窗口和门限为

$$\begin{cases} ssthresh = (BE * RTT \min) / seg_size \\ cwin = 1 \end{cases} \quad (3)$$

从上述TCPW的窗口改动算法中可以看出,TCPW在丢包的时候,并不是盲目地将窗口和慢启动门限减半,而是根据网络的当前情况来设置新的窗口和门限,因此,在随机错误较多的无线链路上,TCPW仍能获得较好的性能。但是正因为TCPW的窗口校准是发生在丢包之后,因此在错误率较低、丢包较少的情况下,TCPW的表现与TCP-NewReno非常相似。尤其在延时较大的环境中,两者的表现都不稳定。

为了考察延时和错误因素对TCPW的影响,本文建立如下的仿真环境对TCPW的性能进行评估,如图1所示。

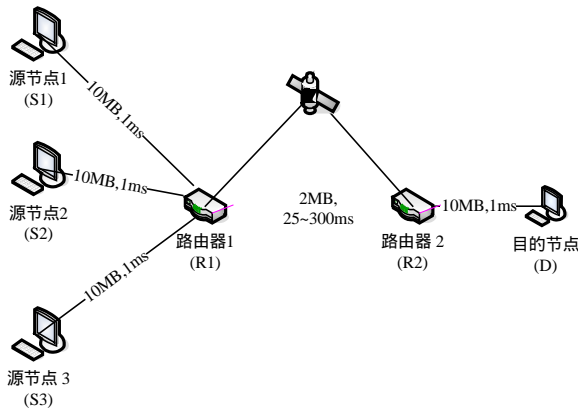


图1 仿真场景

设置3个源节点和一个目标节点通过10MB、1ms的链路和路由器1相连。两路由器之间的链路模拟卫星链路,带宽为2MB,延时在25ms~300ms之间每隔25ms取一个值。该链路上绑定均匀分布的错误模型。为了反应卫星链路资源受限的特性,两路由器之间链路的队列长度仍保持缺省值20。每个源节点和目的节点之间均建立一个TCPW连接。为了充分反应窗口的变化,将3个连接的窗口上限均设为200。

在这种环境下,设置不同的RTT和包错误率PER。3个TCPW流在120s内的总吞吐量和总丢包数如图2所示。可以看出,当延时较小时,一定比例的随机错误对提升TCPW的性能是有益的。随着延时的加大,TCPW的吞吐量迅速下降,但其波动幅度明显增加。这说明,虽然TCPW的拥塞控制策

略较之传统TCP精确了许多,但由于它依靠丢包来调整窗口,系统不能及时收敛到一个稳定状态。

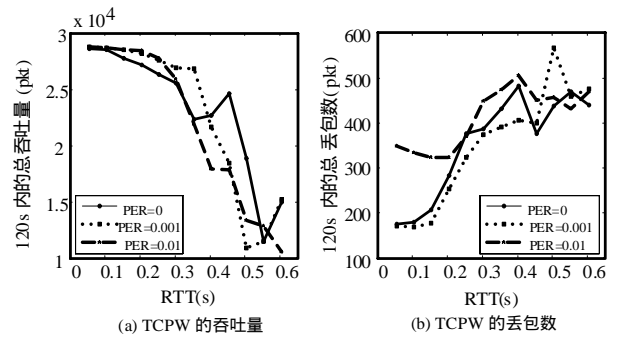


图2 TCPW的吞吐量和丢包数

图3反映了不同RTT时3个连接的总吞吐量。在RTT较小的时候,如图3(a)的情况,每秒的吞吐量迅速达到并基本稳定在瓶颈带宽的饱和值240包/s或241包/s(对应带宽分别是1.9968MB和2.00512MB)。随着RTT的增加,毛刺增多,逐渐过渡到成为图3(c)不能稳定的情况。这主要是因为在大的延时的情况下,上一RTT的带宽估计和当前RTT的可用带宽之间的相关性下降,导致窗口设置的误差增加,引起的拥塞的可能性增加,吞吐量振荡幅度增加,导致总吞吐量的下降。

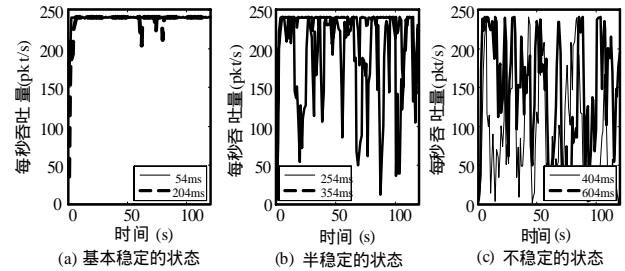


图3 错误率为0时TCPW吞吐量随时间变化图

4 TCPW-S改进方案

卫星的资源受限、分组的丢失和重传将会导致资源的浪费。因此TCPW依靠分组丢失来调整窗口的策略容易造成资源的浪费,或者导致性能的降低。此外,按照第3节的分析,TCPW的拥塞避免算法沿用TCP-Reno的算法,仍较为激进。采用更为保守的窗口调整机制应对TCPW的性能提升和丢包率的下降会更为有益。

按照这种思想,把TCPW的窗口调整与估计带宽紧密联系起来,即使在拥塞避免阶段,也不是简单的线性增加,而是以带宽估计为依据判断窗口的合理性,以避免由于拥塞而导致的分组丢失。在改变窗口的过程中贯彻传统TCP加性增加乘性降低(AIMD)的思想,对窗口改变的幅度进行限制。当测得的可用带宽对应的窗口值大大高于当前拥塞窗口时,保持原有的窗口的增长方式,即慢启动期间窗口按指数增长;拥塞避免期间窗口线性增加,每个RTT窗口增加一个分组。当测得的可用带宽对应的窗口值与当前窗口值的差值在正负一个分组内,就认为当前的拥塞窗口设置是比较合适的,不改变其值。当测得的可用带宽低于当前窗口值,则把窗口设为与当前带宽相一致的值。具体的算法如下:

$$est_win_k = BE_k * RTT \min / seg_size \quad (4)$$

$$cwin'_k = \begin{cases} cwin_{k-1} + 1 & est_win_k > cwin_{k-1} + 1 \text{ 且 } cwin_{k-1} < ssthresh \\ cwin_{k-1} + 1/cwin_{k-1} & est_win_k > cwin_{k-1} + 1 \text{ 且 } cwin_{k-1} \geq ssthresh \\ cwin_{k-1} & |cwin_{k-1} - est_win_k| \leq 1 \\ est_win_k & est_win_k < cwin_{k-1} - 1 \end{cases} \quad (5)$$

$$cwin_k = \begin{cases} cwi n'_k & cwi n'_k \geq 2 \\ 2 & cwi n'_k < 2 \end{cases} \quad (6)$$

在 TCPW-S 的算法中，慢启动门限对窗口变化的影响不大，只有在估计带宽大于当前窗口 + 1 对应的带宽值时，才需要根据慢启动门限确定窗口的增长方式。因在 TCPW-S 对慢启动门限的取值不作更改，仍保持 TCPW 中出现丢包后才更改门限的处理方式。

TCPW-S 在图 1 所示的仿真环境中测试所得的数据见图 4、图 5 所示。图 4 是 TCPW-S 的吞吐量和丢包数。可以看出，窗口的及时降低，大大减小了拥塞发生的可能性，从而提高了吞吐量，降低了丢包数。在包错误率为 0 的实验中，丢包均是拥塞引起的，而拥塞的发生是由于窗口设置不当而导致。因此，考察包错误率为 0 时的丢包数可以看出窗口设置策略是否合适。TCPW-S 在延时较小时(RTT 小于 0.204s 时)丢包数均为 0，而同时吞吐量均高于 TCPW，这证明 TCPW-S 的窗口设置策略要优于 TCPW。由于 TCPW-S 采取了较为保守的窗口改变策略，TCPW-S 的稳定性也要好于 TCPW。从图 5 和图 3 的比较中可以发现，不管 RTT 为多少，TCPW-S 的吞吐量都比 TCPW 更趋近于瓶颈链路的稳定饱和值。

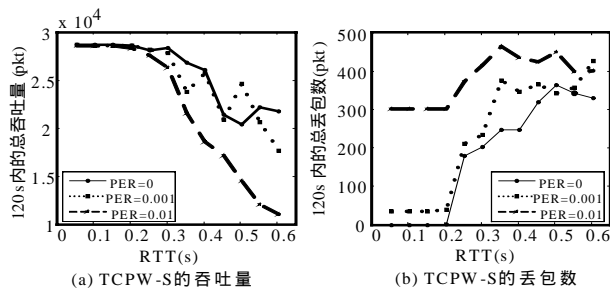


图 4 TCPW-S 的吞吐量和丢包数

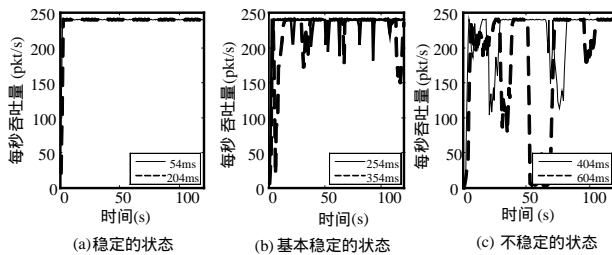


图 5 错误率为 0 时 TCPW-S 吞吐量随时间变化图

TCPW-S 与 TCPW 吞吐量和丢包数的比值如表 1 所示。

表 1 TCPW-S 与 TCPW 吞吐量和丢包数的比值

PE RT	吞吐量比值			丢包数比值		
	0	0.001	0.1	0	0.001	0.1
0.054	1.005	0.998	0.996	0	0.198	0.868
0.104	1.004	0.998	0.998	0	0.202	0.901
0.204	1.051	1.014	0.997	0.007	0.157	0.929
0.304	1.109	1.034	1.016	0.523	0.625	0.926
0.404	1.152	1.189	1.035	0.509	0.852	0.861
0.504	1.084	2.269	1.092	0.830	0.603	0.989
0.604	1.452	1.158	1.047	0.746	0.895	0.855

从 TCPW-S 和 TCPW 的对比分析表 1 中可以看出，TCPW-S 的吞吐量在大部分 RTT 的环境下均好于 TCPW，而且随着延时的增加，其提升幅度也在增加。TCPW-S 的丢包数均好于 TCPW。随着错误率的增加，TCPW-S 的表现逐渐趋近 TCPW，这是由于在丢包较多时，因丢包进行窗口调整的比率大大增加，使两者性能趋于一致。

5 结论

卫星网络是未来 Internet 的重要组成部分，研究适应卫星网特点的传输协议具有重要的意义。TCPW 是针对无线网络而设计的 TCP 改进版本，能够有效地应对无线网络误码率高的特点。但是 TCPW 的性能是以丢包为代价换来的。而且随着延时的增长，TCPW 的性能下降的较快。这两个弱点阻碍了 TCPW 在卫星网络上的运用。

本文在 TCPW 带宽估计算法的基础上，将窗口的适应性调整从丢包后进行改为每 RTT 进行一次。在计算新窗口的时候，本着 AIMD 的原则，限制窗口的上升幅度。一旦带宽估计小于当前窗口，就降低窗口，使其与估算的带宽相一致。仿真实验证明，这种相对保守的窗口改变方式提升了系统的吞吐量，并大幅度地降低了丢包数，节约了宝贵的星上资源。

参考文献

- Mascolo S, Casetti C. TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links[C]//Proc. of MOBICOM. 2001: 287-297.
- Allman M. Ongoing TCP Research Related to Satellites[S]. RFC 2760, 2000.
- Brakmo L S, Peterson L L. TCP Vegas: End to End Congestion Avoidance on a Global Internet[J]. IEEE Journal on Selected Areas in Communication, 1995, 13(8): 1465-1480.
- Akyildiz I F, Morabito G, Palazzo S. TCP-peach: A New Congestion Control Scheme for Satellite IP Networks[J]. IEEE/ACM Trans. on Networking, 2001, 9(3): 307-321.
- Maor A, Mansour Y. AdaVegas: Adaptive Control for TCP Vegas[C]//Proc. of IEEE Global Telecommunications Conference. 2003: 3647-3651.
- Srijith K, Lillykutty J, Ananda A L. TCP Vegas-A: Improving the Performance of TCP Vegas[J]. Computer Communications, 2005, 28(4): 429-440
- Akyildiz I F, Zhang Xin, Fang Jian. TCP-Peach+: Enhancement of TCP-peach for Satellite IP Networks[J]. IEEE Communications Letters, 2002, 6(7): 303-305.
- Kui-Fai L, Kwan L Y. TCP-swift: An End-host Enhancement Scheme for TCP over Satellite IP Networks[C]//Proc. of International Symposium on Computers and Communications. 2004: 551-555.
- Kenshin Y, Wang Ren, Sanadidi M Y, et al. TCP Westwood with Agile Probing: Dealing with Dynamic, Large, Leaky Pipes[C]//Proc. of IEEE International Conference on Communications. 2004: 1070.
- 章 淼, 吴建平, 林 闯. 互联网端到端拥塞控制研究综述[J]. 软件学报, 2002, 13(3): 354-363.